

Study of Load Balancing of Resource Namespace Service

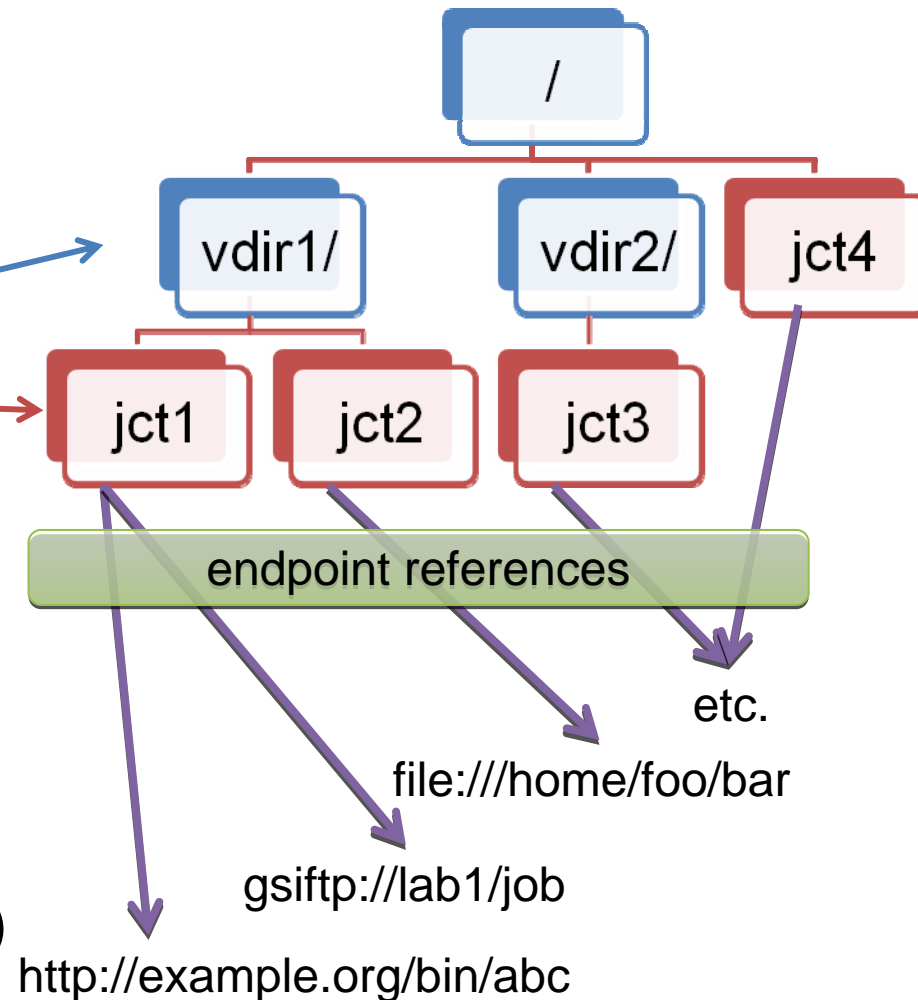
Masahiro Nakamura, Osamu Tatebe
University of Tsukuba

Background

- Resource Namespace Service (RNS) is published as GDF.101 by OGF
- RNS is intended to support an environment of globally distributed cluster of clusters
- Load balancing is required
 - to support requests from possibly more than 100 clients with various latencies
 - to reduce network latency
 - to prevent single-point failure

Resource Namespace Service

- Maps resources to a hierarchal namespace through Endpoint references(EPR)
 - Virtual directory
 - Junction
- Supports following operations
 - add(add entries)
 - list(list entries)
 - update(update entry)
 - query(obtain entry info)
 - remove(remove entries)



Problems in large-scale environment (1)

- Round trip times increase when nodes are located in distant places
 - RTT between Japan-USA is 150ms+
 - Increase of RTT affects response time severely
- Replicating RNS metadata to several locations is required to improve response time

Problems in large-scale environment (2)

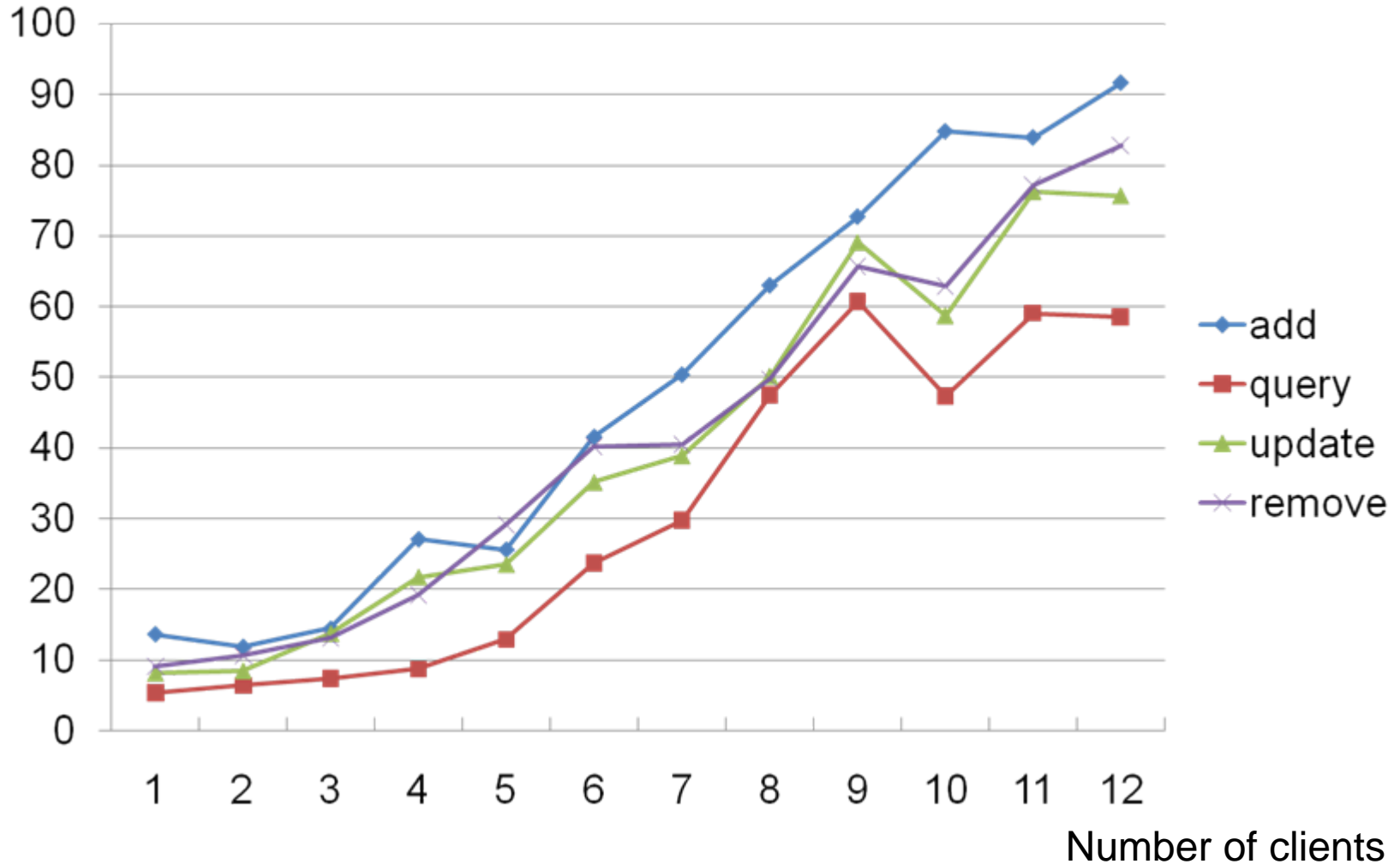
- Evaluate response times of each RNS operation from 1 to 12 clients

Machine configuration

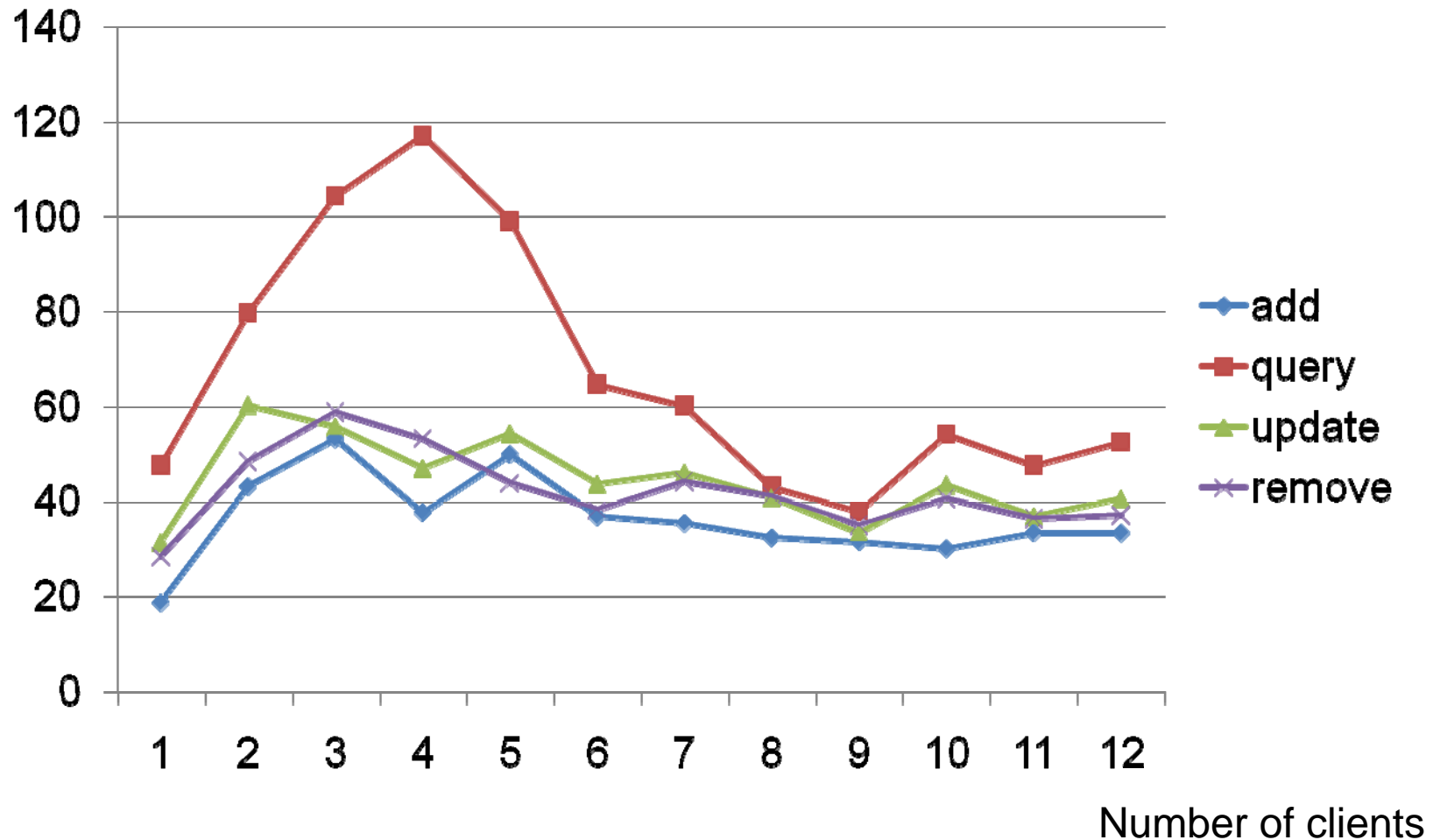
CPU	Intel Xeon 3060 (2.4GHz)
Memory	PC2-5300 2GB
Network	1000BASE-T Ethernet
OS	CentOS 5.2, Kernel 2.6.18 x86_64
Java VM	Java Runtime Environment 6 Update 3 and Java EE 5 Update 3
Hosting environment	Sun Java System Application Server 9.1
Database	MySQL Community Server 5.0.51

Response time per request

Average response time per request (ms)



Operations per second

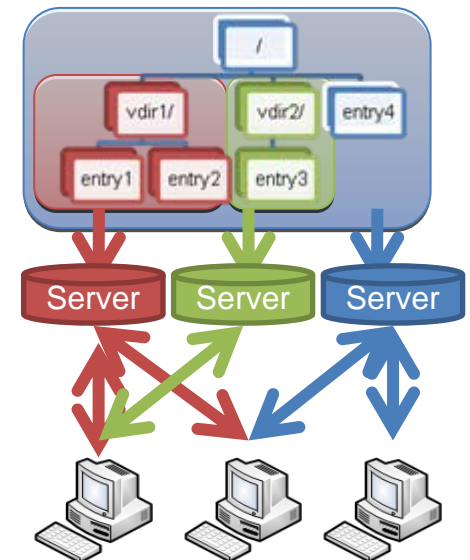
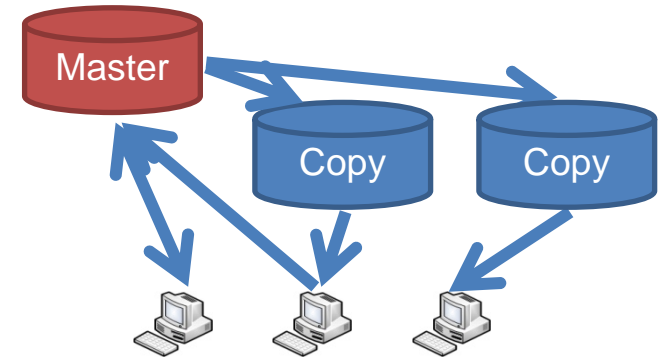


Single server performance

- Response time increases as the number of clients increases
- A server can handle up to 40 – 50 requests/s
 - Exception: ~ 117 query requests/s
 - Not enough when accessed by many clients simultaneously

Load balancing strategies

- Duplicated RNS servers
- Subtree partitioning

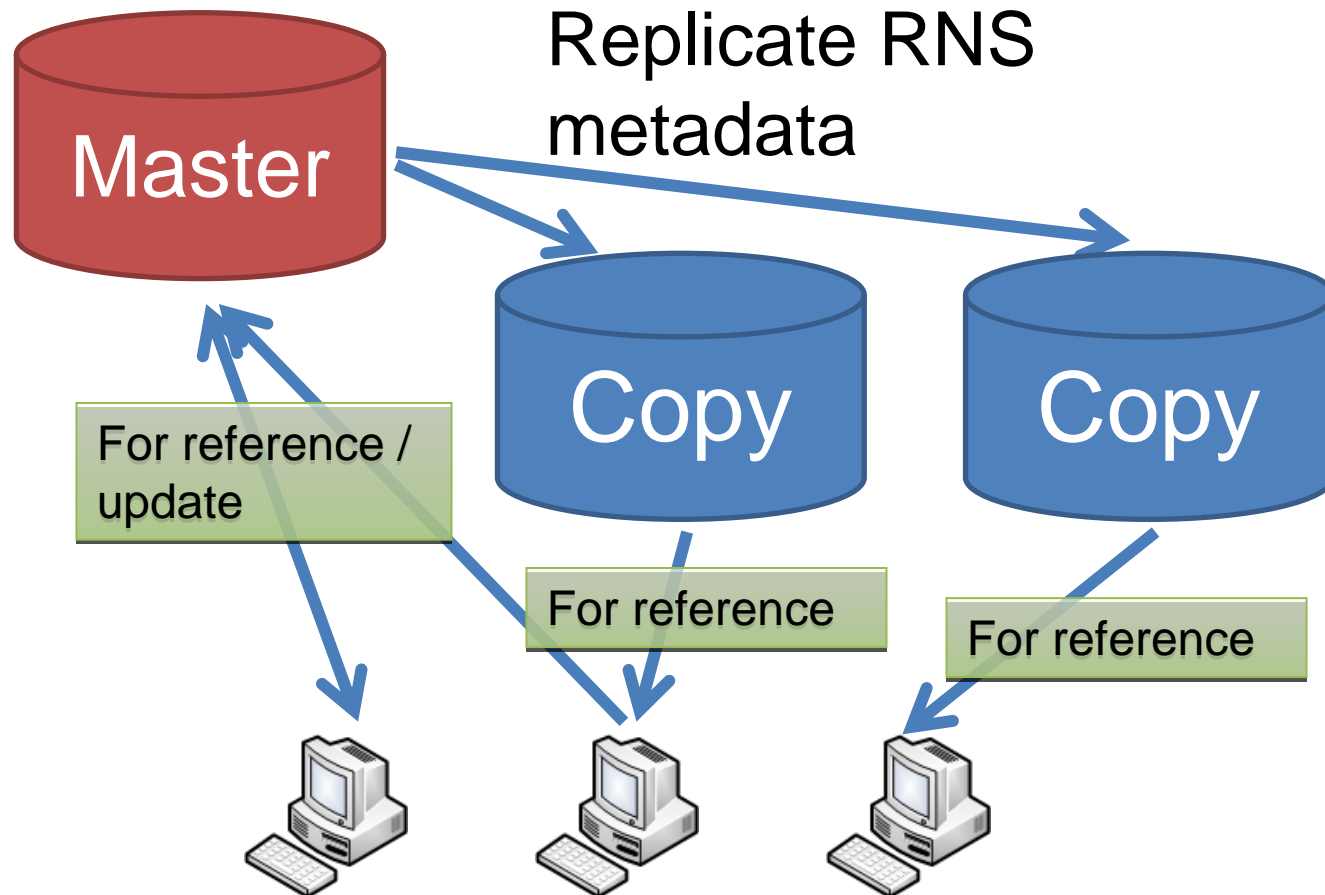


Duplicated RNS servers by primary copy strategy (1)



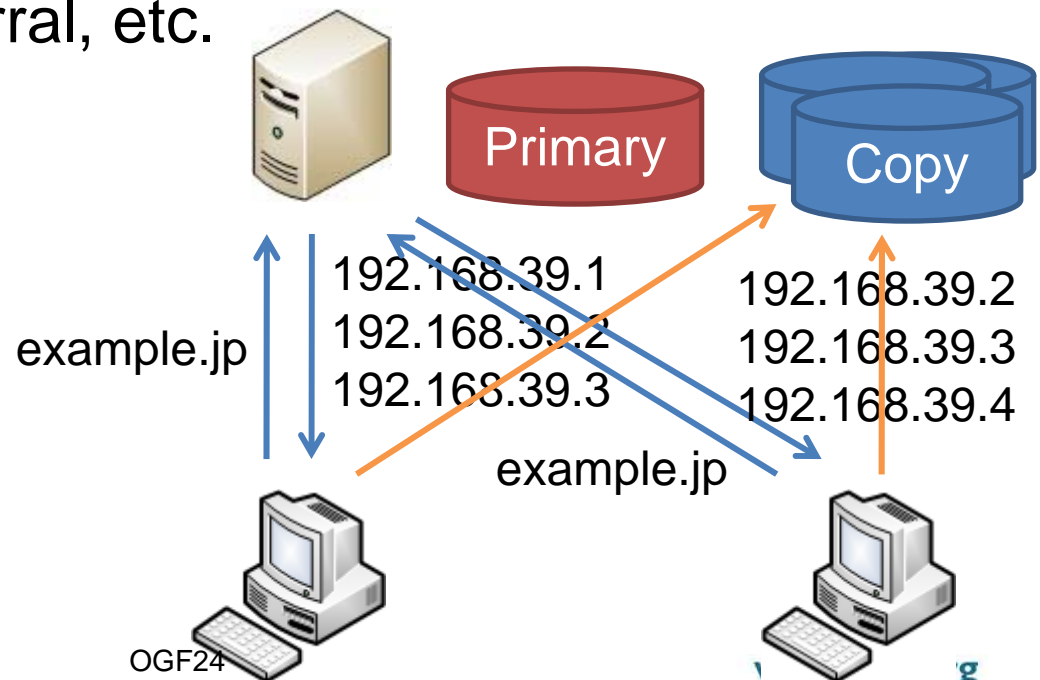
- One master server manages namespace and duplicates NS to other servers
- Clients issue requests to one of the servers to reduce master server load
- All update requests should be issued to the primary server to keep consistency

Duplicated RNS servers by primary copy strategy (2)



RNS server selection

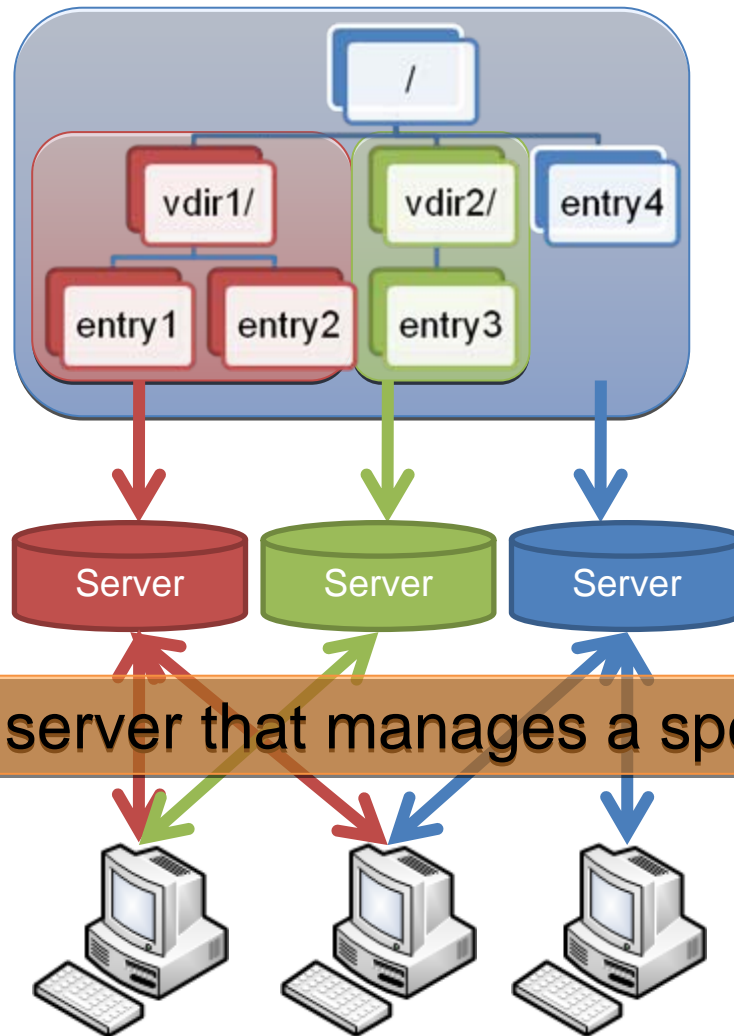
1. Connect to the server pointed to by an EPR
2. Client connection is redirected by
 - DNS round-robin
 - EPR renewal by WS-Naming
 - returning referral, etc.



Subtree partitioning (1)

- Assign subtrees of namespace to different servers
- Consistency is guaranteed if a subtree is not duplicated
- Requests to a single directory cannot be distributed
- Subtree mapping is decided either statically or dynamically

Subtree partitioning (2)



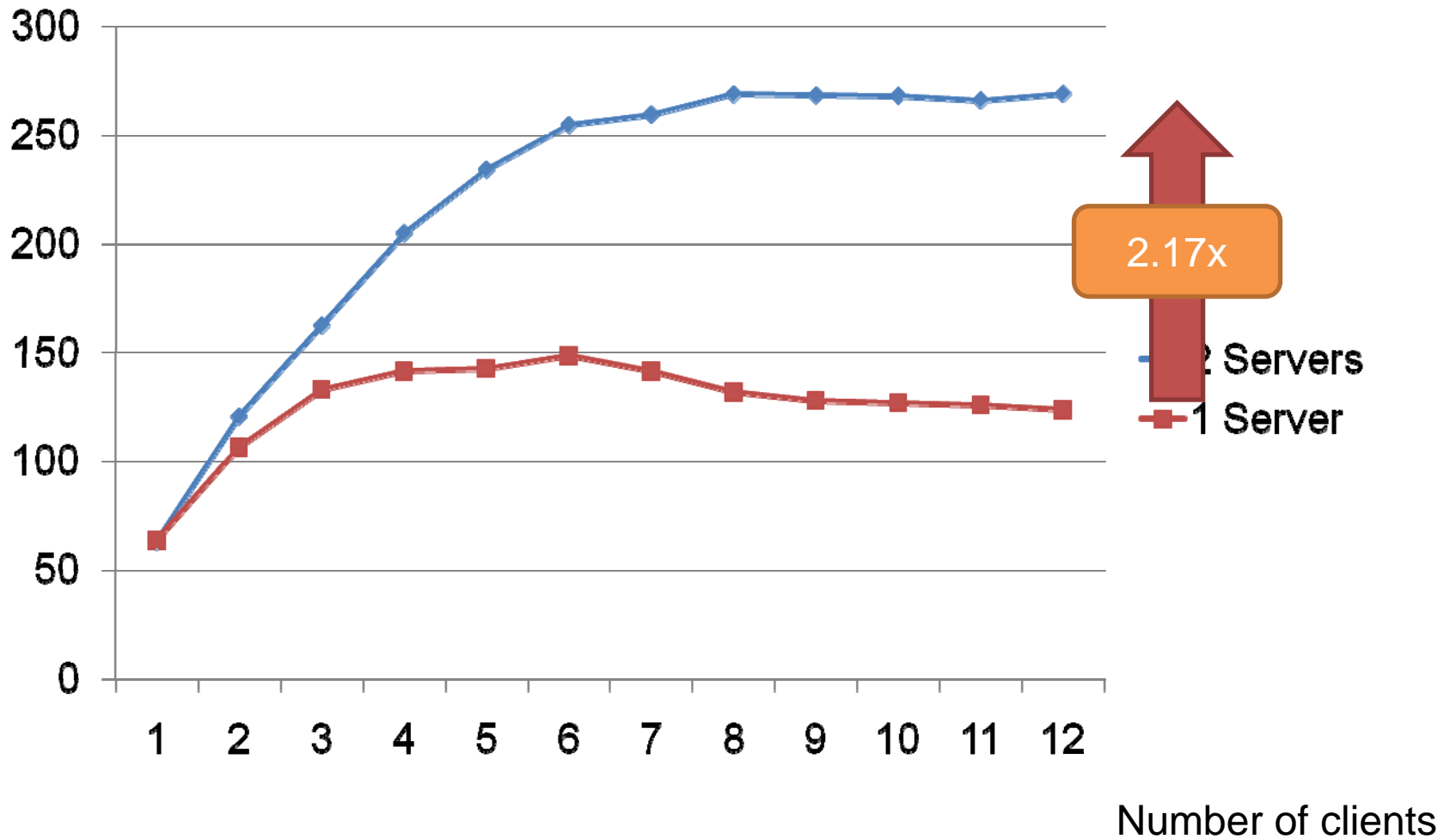
Comparison of load balancing strategies

Strategy	Multiple sites support (references)	Load balancing of update requests	Load balancing of a directory	Dynamic load adaption
Duplicated RNS servers	Good Each site can hold replicas	Poor All updates are redirected to master server	Good	Good
Static subtree partitioning	Poor Requires distant metadata accesses	Good No overhead to keep consistency	Poor Directory is the minimum unit	Poor
Dynamic subtree partitioning	Poor	Good	Poor Same as above	Good

Performance evaluation of duplicated RNS servers (1)

- Measured performance of duplicated RNS servers
- Each client continuously issues query requests
- Requests to duplicated servers are equally balanced

Requests processed per second



Summary of duplicated RNS servers performance



- Throughput improved to 269 requests/s from 123 requests/s
 - 2.17x boost
- Database and/or disk caches may help this super-linear speedup

Summary

- Evaluated duplicated RNS servers, improving performance by 2x with 2 servers
- Further considerations
 - Implement and examine other load balancing methods including subtree partitioning
 - Introduce referral to the RNS standard to provide more choices for server selection